IMPACT OF DYNAMIC VOLTAGE SCALING (DVS) ON CIRCUIT OPTIMIZATION

A Thesis

by

CARLOS ALBERTO ESQUIT HERNANDEZ

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 2009

Major Subject: Computer Engineering

IMPACT OF DYNAMIC VOLTAGE SCALING (DVS) ON CIRCUIT OPTIMIZATION

A Thesis

by

CARLOS ALBERTO ESQUIT HERNANDEZ

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Jiang Hu
Committee Members,	Sunil Khatri
	Duncan M. Walker
Head of Department,	Costas Georghiades

May 2009

Major Subject: Computer Engineering

ABSTRACT

Impact of Dynamic Voltage Scaling (DVS) on Circuit Optimization. (May 2009)

Carlos Alberto Esquit Hernandez, B.S., University of the Valley of Guatemala

Chair of Advisory Committee: Dr. Jiang Hu

Circuit designers perform optimization procedures targeting speed and power during the design of a circuit. Gate sizing can be applied to optimize for speed, while Dual-V_T and Dynamic Voltage Scaling (DVS) can be applied to optimize for leakage and dynamic power, respectively. Both gate sizing and Dual-V_T are design-time techniques, which are applied to the circuit at a fixed voltage. On the other hand, DVS is a run-time technique and implies that the circuit will be operating at a different voltage than that used during the optimization phase at design-time. After some analysis, the risk of non-critical paths becoming critical paths at run-time is detected under these circumstances. The following questions arise: 1) should we take DVS into account during the optimization phase? 2) Does DVS impose any restrictions while performing design-time circuit optimizations?. This thesis is a case study of applying DVS to a circuit that has been optimized for speed and power, and aims at answering the previous two questions.

We used a 45-nm CMOS design kit and flow. Synthesis, placement and routing, and timing analysis were applied to the benchmark circuit ISCAS'85 c432. Logical Effort and Dual-V_T algorithms were implemented and applied to the circuit to optimize

for speed and leakage power, respectively. Optimizations were run for the circuit operating at different voltages. Finally, the impact of DVS on circuit optimization was studied based on HSPICE simulations sweeping the supply voltage for each optimization.

The results showed that DVS had no impact on gate sizing optimizations, but it did on Dual-V_T optimizations. It is shown that we should not optimize at an arbitrary voltage. Moreover, simulations showed that Dual-V_T optimizations should be performed at the lowest voltage that DVS is intended to operate, otherwise non-critical paths will become critical paths at run-time.

To my parents, Angel Esquit and Maria Hortensia Hernandez († Dec. 2000), who have been the greatest parents I could imagine and have supported me in every way possible. To Angel, Ivan, Lorena, Sandra, and the whole family, who have helped me in some or other way through my life and have contributed to my success.

ACKNOWLEDGEMENTS

First of all, I would like to thank my advisor and committee chair, Dr. Jiang Hu, for his support and help throughout the course of this research. I also want to thank my committee members, Dr. Sunil Khatri and Dr. Hank M. Walker for the priceless knowledge acquired through their interesting lectures while being their student here at Texas A&M University.

Thanks to all the staff members at the Department of Electrical and Computer Engineering for helping me with all the procedures and paper work throughout my graduate program.

To my friends, inside and outside the Department of Electrical and Computer Engineering for helping me in a variety of ways academically and at the personal level during my two years living here in College Station.

Special thanks to my friend MSc. Victor Cordero for all the help with the VLSI tools, encouragement and even waking me up early in the mornings to push me into work!, thanks to him also for the brainstormings which shaped the experimental setup of this research project.

Last but not least, I want to thank all the staff members at the office of sponsored students at Texas A&M University, for their invaluable and prompt help with a wide range of issues.

NOMENCLATURE

DVS	Dynamic Voltage Scaling
V_T	Threshold voltage
V_{DD}	Supply voltage
LE	Logical Effort
HDL	Hardware Description Language
PDP	Power-Delay Product
RDV	Relative delay variation

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

1. INTRODUCTION

Power is one of the major concerns in today's VLSI circuit design. Leakage power has become an important source of power consumption, and nowadays its magnitude is comparable to that of dynamic power for large designs and keeps growing exponentially with device scaling [1]. Different techniques have been developed to cope with both dynamic and leakage power consumption. Dynamic Voltage Scaling is a well-know technique for reducing dynamic power, while Dual-V_T is one of the techniques used for reducing leakage power. Traditionally, circuit designers have targeted speed as the variable to optimize, but current VLSI design requires optimizations not only for speed, but for power as well. Optimization for power targets not only to high-performance power-hungry designs, but also to lower-end circuits for low-power consumer electronics so that battery life is extended. Gate sizing is a common technique to optimize for speed, and is often used in combination with low-power techniques such as Dual-V_T. Optimization techniques for speed and power have been extensively researched. The design space for many of these techniques comprises any of the following dimensions: supply voltage (V_{DD}), threshold voltage (V_T), and gate size. Some techniques optimize using only one of those dimensions, while others use either two or three dimensions, targeting solutions closer to the optimum power consumption subject to determined speed requirements.

This thesis follows the style and format of *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems.*

Techniques in [2, 4] use the threshold voltage as the only dimension for power optimization. The design space for [7, 8] comprises supply voltage V_{DD} and threshold voltage V_T. In [5, 6] both threshold voltage and gate sizing are used to perform optimizations. More elaborate algorithms use the supply voltage, threshold voltage and gate sizing as their design space for optimizations as in [3].

In current VLSI design, a sequence of tasks for speed improvement, leakage power reduction, and dynamic power reduction is a typical scenario for circuit optimization. In this thesis we present a case study of the impact of applying Dynamic Voltage Scaling to a circuit that has been optimized for speed and leakage power using a gate sizing technique and a Dual-V_T algorithm, respectively. The remainder of this thesis is organized as follows. Section 2 presents the background concepts that are most relevant to the problem formulation in section 3. In section 4 we present the experimental setup and methodology used to perform the case study. The results are presented and discussed in section 5. Finally, section 6 presents the conclusions and future work.

2. BACKGROUND

2.1. DUAL-VT TECHNIQUES

To understand the advantage of using Dual-V_T techniques, we first need to look at the leakage current of a MOSFET device. (1) expresses the leakage current of a nMOSFET, where I_{ds0} is the current at threshold and is dependent on process and device geometry, *n* is a process-dependent term affected by the depletion region characteristics and v_T is the thermal voltage (26mV at room temperature) [9].

$$I_{ds} = I_{ds0}\, e^{\frac{V_{gs} - V_T}{n v_T}} \left[1 - e^{\frac{-V_{ds}}{v_T}} \right] \tag{1}$$

(1) shows that the leakage current can be reduced exponentially by increasing the threshold voltage V_T. The leakage power expressed in (2) is also reduced exponentially.

$$P_{Leakage} = I_{ds}\ \ V_{DD} \tag{2}$$

Next we need to look at the simple α ‑ power model for the delay of a CMOS gate [8]:

$$T_d \approx \frac{C_L \cdot V_{DD}}{K\,(V_{DD} - V_T)^{\alpha}} \tag{3}$$

Where C_L is the load capacitance, *K* is a factor that depends on the process and the gate size, and α takes any value between one, for full velocity saturation, and two, for no velocity saturation as in long channel devices [8]. (3) shows that increasing V_T produces an increase in the delay T_d of the gate. Dual-V_T algorithms work based on (1) and (3). After synthesis, circuits have a path distribution similar to the one shown in Fig. 1. The traditional approach to V_T selections relies on the observation that a circuit's

overall performance if often limited by a few critical paths [10]. Many paths are unnecessary faster than the critical path(s). The concept of Dual-V_T algorithms is to slow down those paths by increasing the threshold voltage V_T in as many gates as possible, as long as the paths don't become slower than the original critical path(s). All gates in critical path(s) are kept with low-v_T so that the maximum frequency of operation is not altered. All gates with increased v_T will have less leakage current, according to (1), thus reducing the total leakage power of the circuit without any performance penalty.

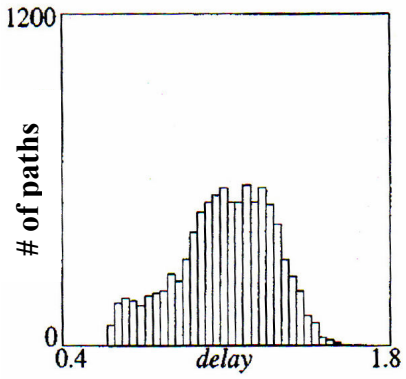

Fig. 1. Path delay distribution of a circuit after synthesis [10]

Assuming that we have a circuit that has already been optimized for speed, the idea of applying a Dual-V_T algorithm is to equalize the delays of all the paths in the circuit as much as possible by assigning high-v_T gates in all non-critical paths. If all gates in the circuit are assigned to high-v_T (single high-v_T) then the critical path of the circuit would become larger than the original before high-v_T assignments. One important characteristic is that Dual-V_T CMOS has the same critical delay as the single

low-v_T circuit [11], but consumes less leakage power due to the gates reassigned to high-vt in non-critical paths. Fig. 2 shows an example of this characteristic for a 32-bit adder.

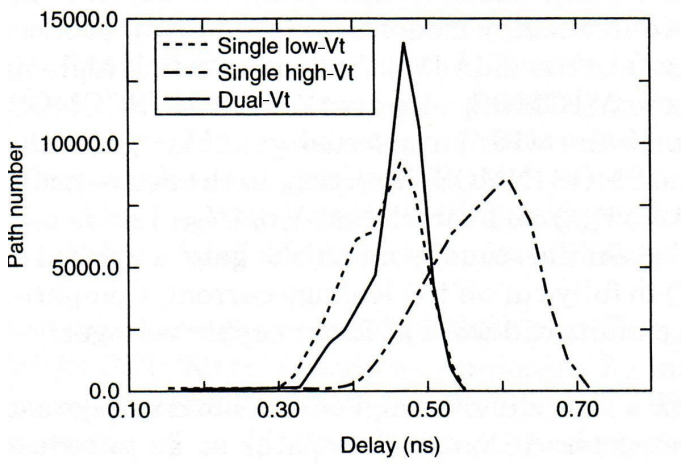

Fig. 2. Path distribution of dual V_T and single V_T CMOS [11]

2.2. DYNAMIC VOLTAGE SCALING

In addition to leakage reduction, Dynamic Voltage Scaling (DVS) is commonly used in microprocessor design to achieve substantial dynamic power reduction. DVS allows devices to dynamically change their speed and voltage, increasing the energy efficiency of their operation [12]. (4) expresses the dynamic power consumption for a CMOS gate, where α is an activity factor, C is the capacitance being switched and *f* is the operating frequency. Reducing the supply voltage V_{DD} will offer a quadratic reduction in dynamic power. At the same time, from (3), reducing the supply voltage will also increase the delay of the gate, this implies that the frequency of operation has to be slowed down in order to avoid setup-time violations.

$$P_{dynamic} = \alpha C V_{DD}^2 f \quad (4)$$

Slowing down the frequency without voltage scaling is not useful from an energy-efficient viewpoint, since the power savings are offset by an equal increase in execution time [14]. The idea of DVS in a microprocessor is to vary the supply voltage under software control to meet dynamically varying performance requirements [13]. The goal of DVS is to save power by reducing the performance of the processor without causing an application to miss its deadlines, this concept is shown in Fig. 3, in A the workload runs at full speed and finishes well in advance of its deadline. In B, the execution of the workload is stretched to its deadline, which allows for energy savings on processors that implement voltage scaling. Completing a task before its deadline and then idling is less energy efficient than running the task more slowly to begin with, and meeting its deadline exactly [14].

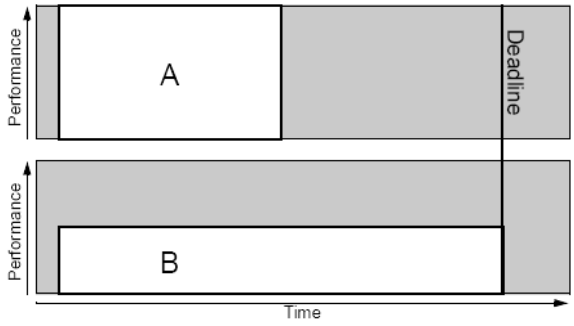

Fig. 3. Performance scaling [14].

The implementation of DVS requires algorithms, termed *voltage schedulers* to determine the operating speed of the processor at runtime. Different voltage schedulers have been compared in [13]. Basically, voltage schedulers work at the operating system level and determine how much the operating frequency can be slowed down while still

meeting performance requirements. Voltage schedulers control the processor speed by writing the desired clock frequency to a control register. The register's value is used by a control loop on-chip to adjust the CPU clock frequency and regulated voltage [15]. A voltage-controlled oscillator (VCO) in the control loop generates a clock signal f_{CLK} which is proportional to the critical path delay of the core over process and temperature changes [16]. Fig. 4 shows an example of a control loop used by DVS. Something very important to mention is that most DVS control loop designs use an inverter-based delay chain or ring oscillator to model the critical path of a circuit. It has been shown that the delay of an inverter-based ring oscillator should accurately track the delay of the critical path as shown in [17]. The delays do not need to be exactly the same, as long as they track each other. A gain factor in the control loop can adjust the delay of the ring oscillator to match the delay of the critical path.

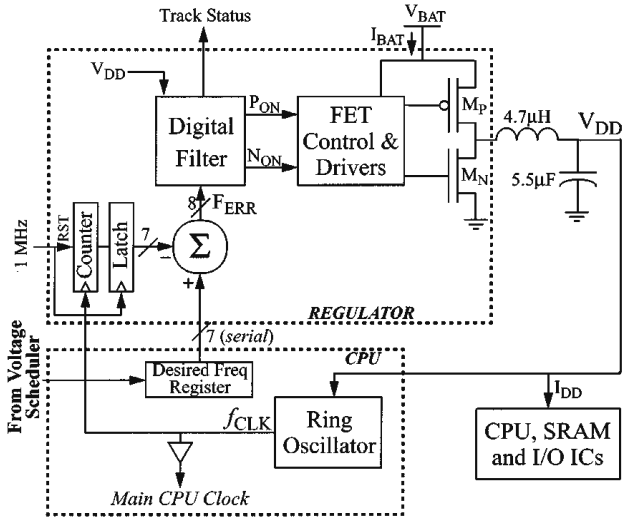

Fig. 4. Frequency to voltage feedback control loop for DVS [15]

2.3. LOGICAL EFFORT AND GATE SIZING

Logical effort is a design procedure for achieving the least delay along a path of a logic network [18]. The method of logical effort is founded on a simple model of the delay through a single MOS logic gate. The model describes delays caused by the capacitive load that the logic gate drives and by the topology of the logic gate. All optimizations with this method are performed by gate sizing. Logical effort is a powerful tool for speed optimization due to its effectiveness and simplicity; these characteristics make it a good option for use in very different scenarios. In [19], logical effort is used to optimize for speed in a synthesis algorithm. The method has also been used in software tools as in [24]. A section about logical effort and transistor sizing is presented in [9]. All the details about the algorithm and its theory are presented in [18].

In a design flow, a method like logical effort would typically be applied after synthesis to optimize for speed, also referred to as *size optimization*. By doing so, the path distribution would change from the one shown in Fig. 1, to a taller and narrower distribution, both distributions are shown in Fig. 5 for ease of comparison.

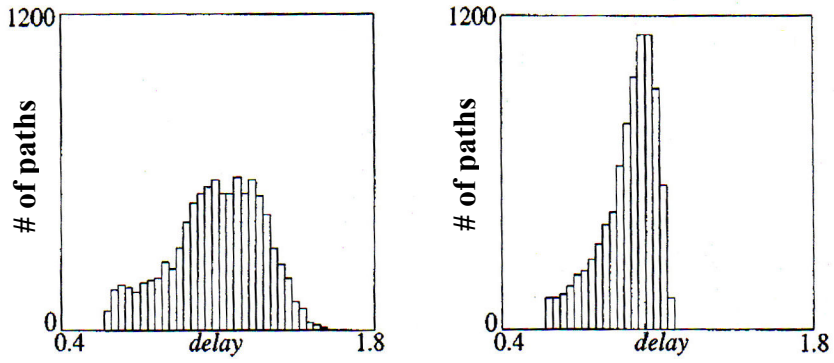

Fig. 5. Path delay distribution of a circuit before and after size optimization [10]

3. PROBLEM FORMULATION

Let us study a VLSI design scenario where a circuit is first optimized for speed using the method of logical effort for gate sizing. The speed-optimized circuit is then optimized for leakage power using a Dual-V_T algorithm. Finally, dynamic voltage scaling (DVS) is used for dynamic power reduction at runtime. Both logical effort and Dual-V_T are design-time techniques, which are applied to the circuit at a fixed voltage. On the other hand, DVS is a run-time technique and implies that the circuit will be operating at a different voltage than that used during the optimizations with logical effort and Dual-V_T. From (3) we see that the delay of a CMOS gate is a function of the size of the gate, the supply voltage, and the threshold voltage. As mentioned in section 2, the delay measurement in feedback loops used in DVS circuits works based on the assumption that circuit delays track well over voltage as shown in [16, 17, 20]. The delay measurement of the critical path is based on that assumption no matter whether the feedback loop uses a ring oscillator or an actual replica of the critical path. But the work in [15] shows that small delay tracking variations exist over voltage which impact circuit timing. In [15] three chains of inverters were simulated whose loads were dominated by gate, interconnect, and diffusion capacitance respectively. To model paths dominated by stacked devices, a fourth chain was simulated consisting of four pMOS and four nMOS transistors in series. The baseline reference is an inverter chain with a balanced load capacitance similar to the ring oscillator. With this setup, 4 structures modeling 4 different types of paths were simulated. As Fig. 6 shows, there is relative delay variation

(RDV) for different path structures over voltage, especially at low voltages when V_{DD} is close to V_T. The range of V_{DD} between V_T and $2V_T$ is particularly interesting to our study since we are going to be operating in that range as it will be explained and justified in section 4.

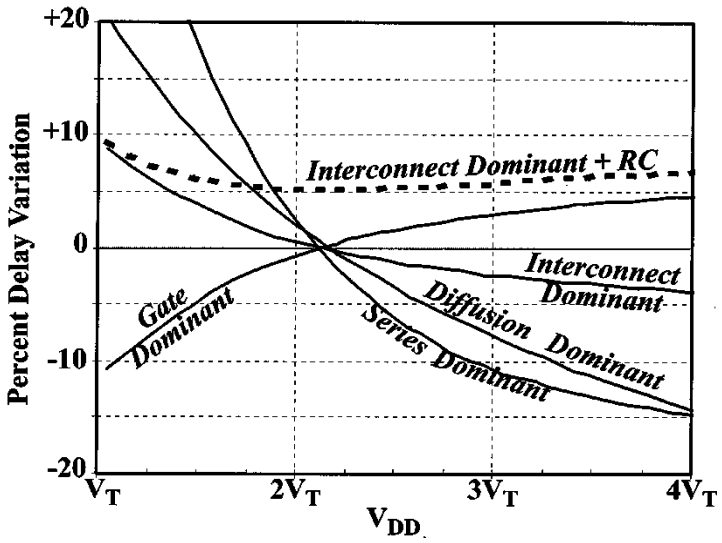

Fig. 6. Relative CMOS circuit delay variation (simulated) [15]

Fig. 6 shows a potential risk for circuits using DVS with voltages operating close to the threshold voltage V_T. The risk is that non-critical paths could become critical paths due to the relative delay variation during voltage scaling at run-time. This situation presents a problem to DVS itself, since the control loop for regulating the supply voltage is based on delay measurements of the critical path, which would be undetermined at runtime. The on-chip replica of the critical path would just not replicate the 'new' critical path. The risk is even higher after size optimization has been performed because the path delay distribution is narrower as shown in Fig. 5.

Although Fig. 6 is showing that relative delay variations between paths do exist, we don't expect those to be large in this case study since the paths in our circuit (ISCAS-85 c432) are mostly all gate-dominant (not many different path structures in the circuit that could scale at considerable different rates).

In addition to the relative delay variation risk just discussed, optimizing using a Dual-V_T algorithm presents another risk, with the same final consequences. A circuit will have a set of high-v_T gates and a set of low-v_T gates after Dual-V_T optimization. We need to calculate the partial derivative of (3) to show the risk.

$$\frac{\partial}{\partial V_T}(T_d) = \frac{\partial}{\partial V_T}\left[\frac{C_L \cdot V_{DD}}{K (V_{DD} - V_T)^{\alpha}}\right] = \frac{\alpha \cdot C_L \cdot V_{DD} (V_{DD} - V_T)^{\alpha - 1}}{K (V_{DD} - V_T)^{2\alpha}} \qquad (5)$$

(5) is expressing the rate at which the delay of a CMOS gate scales with respect to the threshold voltage V_T. Let us express (5) relative to the delay of the gate T_d itself so that the dependence of the rate of scaling on V_T is more clear:

$$\frac{\frac{\partial}{\partial V_T}(T_d)}{T_d} = \frac{\alpha}{(V_{DD} - V_T)} \qquad (6)$$

From (6) we can see that the partial derivative of the delay of a gate with respect to the threshold voltage (and normalized to the delay of the gate T_d) is still a function of the threshold voltage, which means that while applying voltage scaling due to DVS, the delay of gates assigned to high-vt will scale differently than those assigned to low-vt. In

fact, (6) states that high-v_T gates will scale faster than low-v_T gates. Applying a Dual-V_T algorithm with the purpose of minimizing leakage power results in a distribution with many paths very close to the critical path. Again, the risk is that non-critical paths could become critical paths during voltage scaling at run-time.

This thesis is a case study to determine whether this risk exists in a real, practical deep sub-micron VLSI implementation. If the risk exists, DVS should be taken into account during design-time optimizations and would probably restrict those.

4. EXPERIMENTAL SETUP

The case study used the benchmark circuit ISCAS-85 c432. From the viewpoint of the problem formulated in section 3, three fundamental tasks need to be performed: Optimize for speed, optimize for leakage power, and perform DVS to study the behavior of the circuit. The whole design flow used to perform those tasks is shown in Fig. 7.

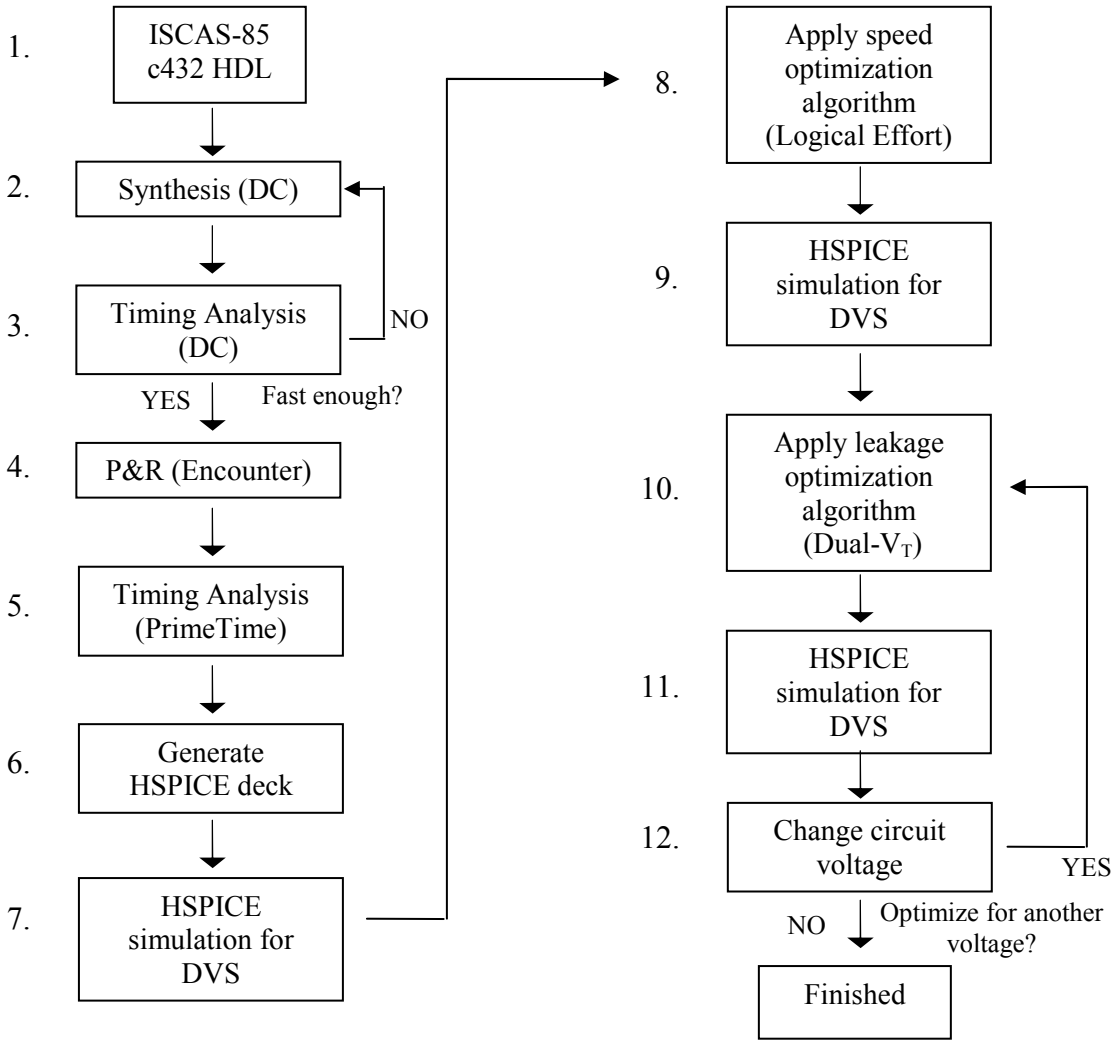

Fig. 7. Design flow used for the case study

4.1. FROM HDL TO LAYOUT

The starting point in the flow of Fig. 7 is a Hardware Description Language (HDL) version of the circuit. We used the Verilog file downloaded from [25]. Synthesis and place & route are performed based on the *FreePDK45nm* design flow jointly developed at Oklahoma State University and North Carolina State University. The flow offers all the libraries and scripts necessary to perform synthesis using *Synopsys Design Compiler* and place and route using *Cadence Encounter*. Libraries for static timing analysis using *Synopsys PrimeTime* are also included. The design flow and libraries can be downloaded from [26]. We performed several iterations of steps 2 and 3 from Fig. 7 in order to get a synthesized circuit fast enough for this technology. Since this was the first time we were using a 45nm flow, the first synthesis was performed with arbitrarily relaxed timing constraints, so that synthesis success was guaranteed and we could get a starting point for re-synthesizing with more realistic timing constraints. A few iterations were performed until the software tool was not able to get a synthesized design. The maximum frequency achieved was 700 MHz. After accepting 700 MHz as our base frequency, we proceeded with step 4 in Fig. 7, that is place & route using *Encounter*. *Synopsys PrimeTime* was used in next step, for static timing analysis. *PrimeTime* is more powerful and precise than *Design Compiler* in the area of static timing analysis, and after the first iteration of place & route we found out that the frequency of operation of the circuit was 800 MHz after parasitics back-annotation, so that no additional P&R iterations were performed. Synthesis and P&R procedures are automated based on the scripts provided by [26]. A timing report was generated using *PrimeTime*, which

contains information of 15 distinctive paths. That is the set of paths used for all simulations and measurements in this case study. A HSPICE netlist was also generated from the final placed & routed design using *Encounter*. The timing report and the HSPICE netlist are the base for all the following steps in the flow of Fig. 7 (steps 7 to 12).

4.2. INITIAL CIRCUIT AND DVS RANGE

A first set of HSPICE simulations is run after place and route to determine the circuit's performance. This set consists of simulations for a supply voltage sweep through the range of DVS. The results from these simulations are the baseline for comparisons after speed and power optimizations.

The cell library downloaded from [26] uses a voltage of 1.1V. All supporting files, like the library file for use with *Design Compiler* and timing library for *PrimeTime*, are based on cell characterizations at that voltage. According to [21], the maximum allowable voltage that can be applied to the gate thickness used in this thesis (1.1 nm), such that no more than 100 ppm oxide breakdown rate in 10 years occurs, is between 1.0V and 1.2V (extrapolated), therefore the 1.1V was chosen as the upper bound for DVS operation. The lower bound for DVS is less critical, since at the end it is about trading off speed for power. The zero-bias threshold voltage for nMOS transistors in the model files from [26] is 0.42 V, and even a higher value is used when applying Dual-V_T optimizations (as presented in section 4.5). Under these circumstances, an arbitrary lower bound of 0.7 V was chosen for DVS.

4.3. AUTOMATION TOOL FOR HSPICE SIMULATIONS

A tool written in C++ was implemented to automate some tasks related to HSPICE simulations to be used from this point (step 7 in the whole flow) to the end of the flow of Fig. 7. The inputs to this tool are the original HSPICE deck generated by *Encounter* and the timing report generated by *PrimeTime*. The tool reads all inputs and outputs from the HSPICE deck of the circuit, then reads all paths from the timing report, and generates decks for power measurement and delay measurement for all paths included in the report. A master HSPICE deck is used as a template to put all decks together by using '.include' statements so that simulations can be run using this master deck.

In the case of power measurements, the tool provides parameters for choosing the number of patterns wanted to use for power measurement and the frequency at which those patterns want to be applied. The tool then generates a HSPICE deck with the random vectors and connects them to the main circuit. The template deck contains the '.measurement' statement for power so that it is ready to be simulated and get power measurement results.

In the case of path delay measurements, the tool reads all the paths from the *PrimeTime* report and generates a HSPICE deck with all the signals required to sensitize all paths, it also generates all the corresponding '.measure' statements. The frequency parameter is used to determine the timing for applying patterns to measure the delay of each path. By using the tool all power measurement patterns, and all input signals to sensitize paths are automatically generated and connected at the deck level. All

'.measurement' statements are automatically generated as well. Then all what is needed to do is to run one simulation for the master power measurement deck and one simulation for the path delay master deck. A Perl script is used to read the information from the .mt0 file generated by HSPICE, and generates a comma-separated value (.csv) file ready to be imported by any standard spreadsheet tool.

4.4. SPEED OPTIMIZATION

Speed optimizations were performed using the method of logical effort presented in [18]. The method was not applied manually, but rather it was automated by means of another tool written in C++. The inputs to the tool are the original HSPICE deck and two *PrimeTime* reports; the same timing report as in the previous tool, plus a complete nets report generated using the 'report_net' tcl command in a *PrimeTime* script. The output from the tool is a new HSPICE deck which consists of the old deck with resized gates that optimize for speed according to [18]. The tool works on single paths, so that the user selects both the start and end points of the path. The tool looks for the specified path in the timing report and generates the optimized HSPICE deck. An output text file is also generated, which contains information about all the resized gates and detailed information about final and intermediate results from the method of logical effort.

No additional details or pseudo-code about the optimization algorithm will be included here, since the algorithm used for speed optimization in this thesis is a precise and straightforward implementation of the method presented in [18]. All the details and theory about the algorithm are presented in [18].

HSPICE simulations sweeping the supply voltage are run after logical effort optimizations to evaluate the impact of DVS on the circuit.

4.5. LEAKAGE OPTIMIZATION

A Dual-V_T optimization tool was implemented based on [2]. The tool was written in C++. The inputs to the Dual-V_T optimizer are the *PrimeTime* timing report and a single-V_T HSPICE deck, which for our flow is simply the output from the speed optimizer. The output is a HSPICE deck using gates mapped to a Dual-V_T version of the library provided by [26]. This Dual-V_T version was implemented by creating a high-v_T version of the MOSFET model cards and a Dual-V_T version of all the cells, which map to the high-v_T MOSFET models. It is important to mention that this thesis is not proposing a leakage optimization algorithm. The tool was implemented with the objective to help automate and integrate the leakage optimization tasks to our flow. The tool was designed to perform Dual-V_T leakage optimization to all paths included in the timing report, not to the whole circuit. This is valid and sufficient for the purposes of this study since all measurements are based on the same set of paths.

The pseudo-code for the Dual-V_T algorithm used for leakage optimizations is shown in Fig. 8. For the selection of the high-v_T value, the rule of thumb in [22] was used. The rule of thumb states that the difference between high-v_T and low-v_T should be approximately 0.1 V so that the leakage reduction from Dual-V_T is optimal. The key concept behind the rule is that if the difference between the high and low threshold voltages is too large, the leakage reduction achieved by using each high-v_T gate is high,

but just a few gates can be assigned to hight-v_T. On the other hand, if the difference between the two threshold voltages is too small, many gates can be reassigned to high-v_T, but the leakage reduction achieved by each gate will be small.

```
High-V_T Assignment
{
   Create a copy of the input HSPICE deck (temp deck)

  For each path i
   {
      For each gate j
      {
            Assign high-v_T to j in temp HSPICE deck
            Launch  temp deck HSPICE simulation and measure the slacks
            δ_1 … δ_n at  all outputs 1 … n
            If  ( δ_m ≥ 0, for 1 ≤ m ≤ n )
                  keep j at high-v_T  in temp deck
            Else
                  Reassign j to low-v_T in temp deck
      }
   }

  Write temp HSPICE deck to output HSPICE deck
}
```

Fig. 8. Pseudo-code for the Dual-V_T optimizer implemented in C++

The circuit was set to operate at the lowest voltage in the DVS range (0.7V). The first Dual-V_T optimization was run at this voltage, corresponding to step 10 in our flow. Then HSPICE simulations are run sweeping the supply voltage through the whole DVS range. The master HSPICE deck file described in section 4.3 is used to automatically sensitize all the paths and measure all the corresponding delays. After this, the voltage is set to a different value in step 12, and a loop is performed by going back to step 10 in

our flow. Each iteration at a different voltage is used to study the impact of DVS on different circuit optimizations (Dual-V_T optimizations at different voltages). The loop was iterated 3 times, for gathering data of the circuit operating at 0.7V, 0.9V and 1.1V. Those voltages correspond to the lowest, middle, and highest values within the DVS range.

5. RESULTS

The case study flow of Fig. 7 was applied to the benchmark circuit ISCAS-85 c432 using the 45nm design kit provided by Oklahoma State University and North Carolina State University [26]. The *PrimeTime* timing library in [26] needed to be recompiled due to compatibility issues with software versions currently installed on the electrical engineering servers at Texas A&M University. Subsections 5.1 and 5.2 will first show the results validating the optimization algorithms used for speed and leakage power. The results for the impact of applying DVS will be presented in section 5.3.

5.1. SPEED OPTIMIZATION RESULTS

The algorithm of logical effort was applied to the four most critical paths out of the set of 15 paths used in this case study. Optimization for all other paths was not necessary since all of them were shorter than the most critical path even after speed optimization. A circuit speedup between 14% and 17% was achieved though the whole DVS range. Fig. 9 shows the delay of the four most critical paths before any optimization. Fig. 10 shows the delays after speed optimization, normalized to the original delays after place and route. Fig. 11 shows the circuit speedup achieved for the DVS range. It is shown that circuit speedup decreases for increasing supply voltage; this is mostly due to gate delay scaling without corresponding wire delay scaling. The tool implemented according to the method of logical effort optimizes for speed by gate sizing, but does not perform any interconnect delay optimizations. The larger the supply

voltage, the smaller the gate delay, but the wire delay remains constant. The overall circuit speedup is then smaller for larger supply voltages.

The method of logical effort explains how to design a path for maximum speed, but does not easily show how to design a path for minimum area or power under a fixed delay constraint [18]. We are not targeting the minimum-energy design, since to achieve that we should follow a three-dimension design space procedure, optimizing for supply voltage, gate sizing and threshold voltage as in [3]. A procedure like that doesn't take into account DVS. This case study uses logical effort for speed optimization only, without any power constraints. Fig. 12 and Fig. 13 show the speed-power tradeoff due to logical effort.

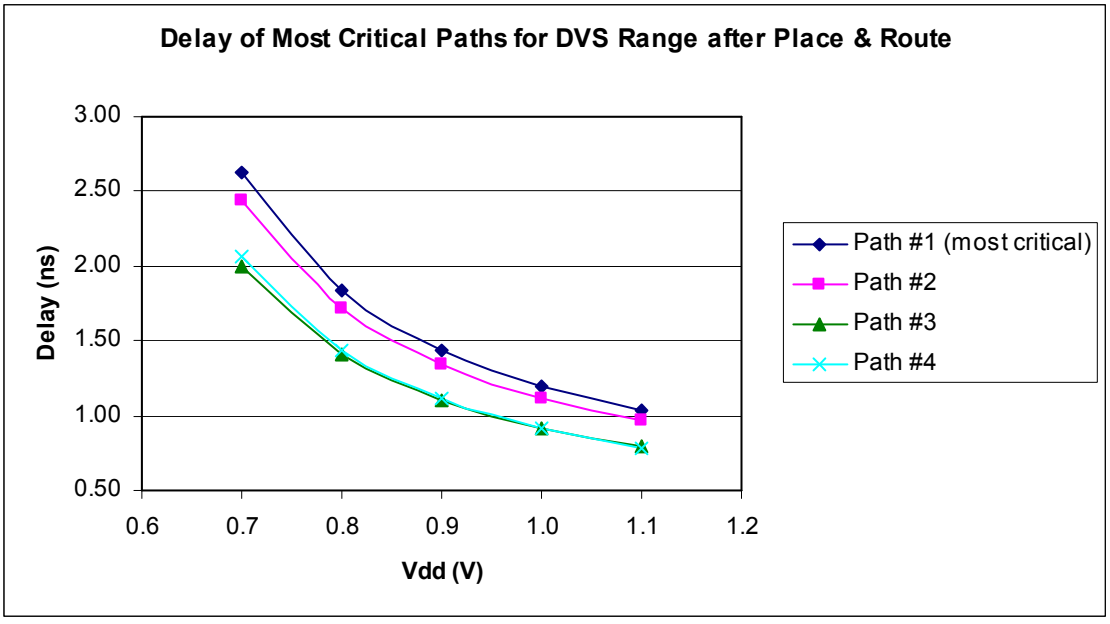

Fig. 9. Delay of the four most critical paths after place & route

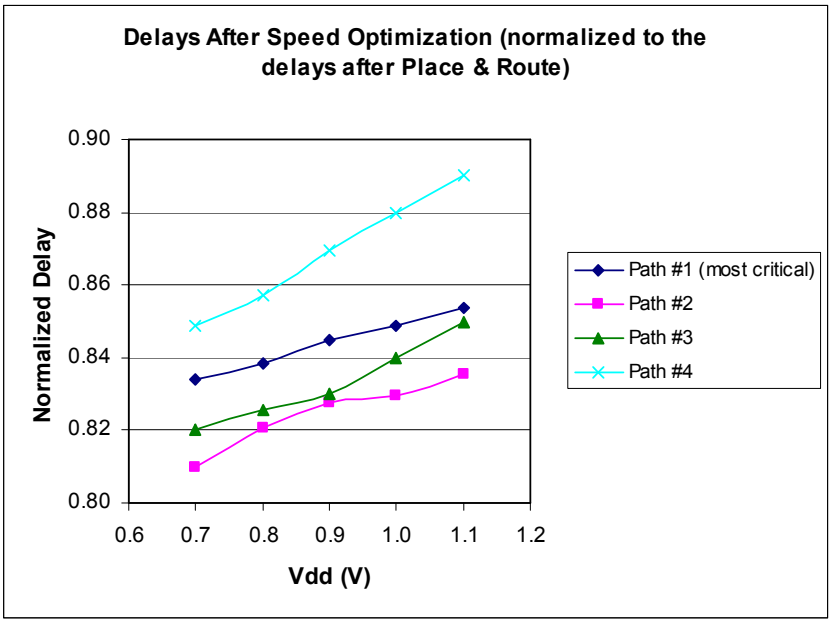

Fig. 10. Delays after speed optimization, normalized to the delays after place & route

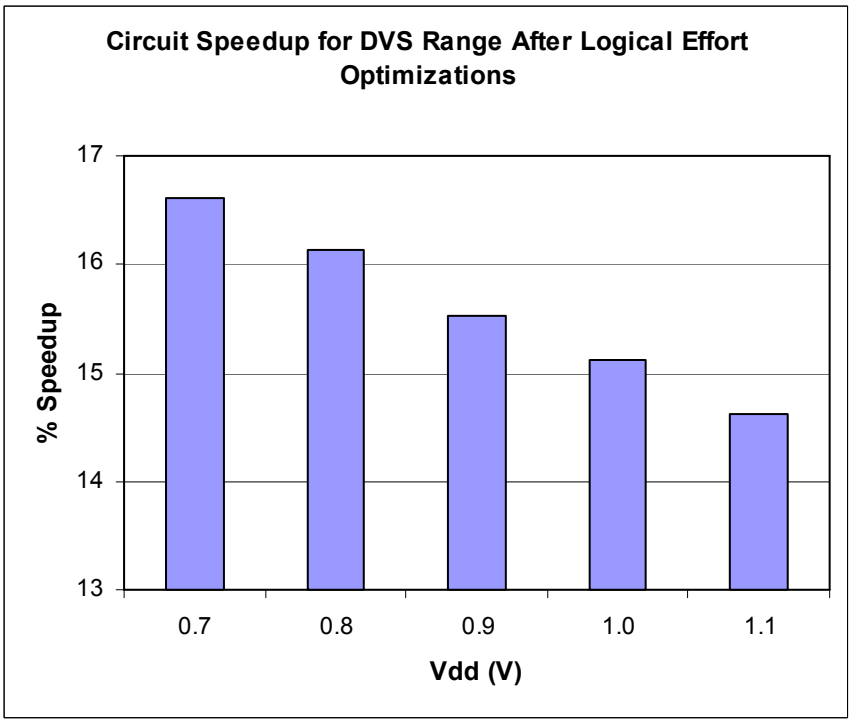

Fig. 11. Circuit speedup for the whole DVS range after LE optimization

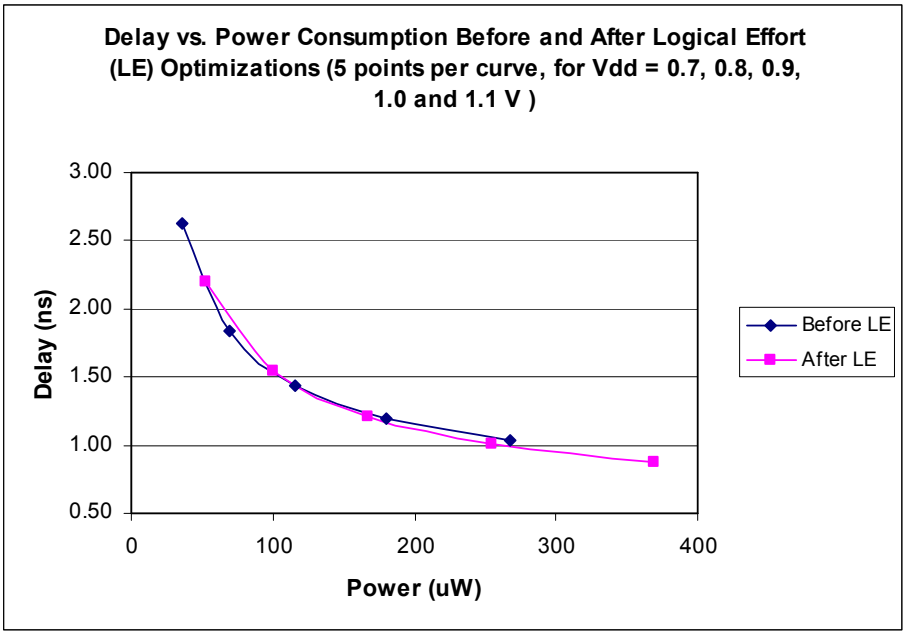

Fig. 12. Delay vs. power before and after speed optimizations

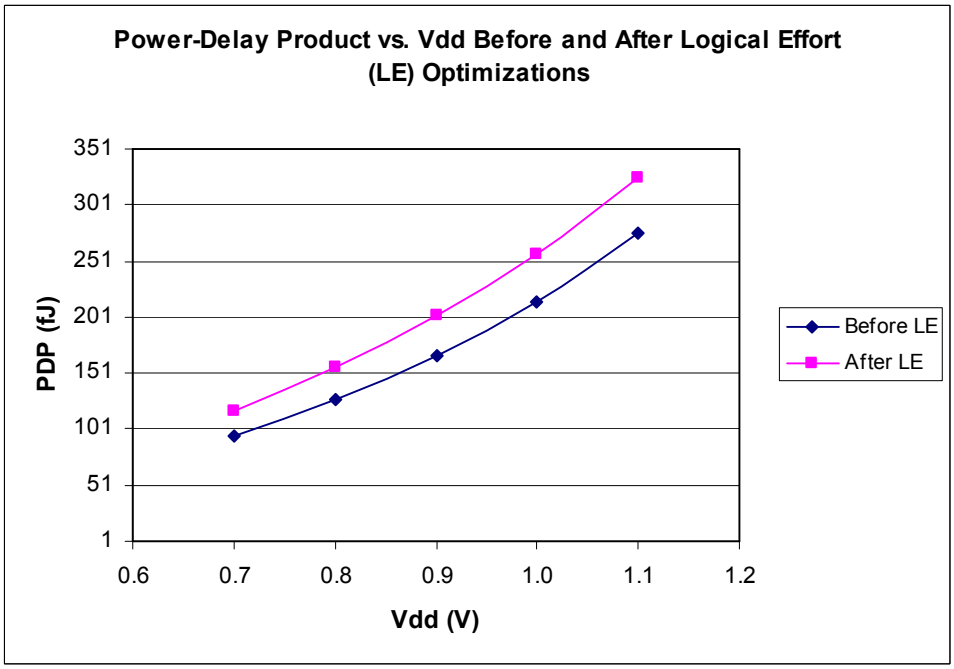

Fig. 13. Power-Delay Product before and after speed optimizations

5.2. LEAKAGE OPTIMIZATION RESULTS

Dual-V_T optimizations were run for the circuit operating at the lowest, middle, and highest voltages within the DVS range (0.7V to 1.1V). Fig. 14 shows the leakage power savings after applying the Dual-V_T optimization tool implemented in C++. Power savings can be as large as 52% when operating at a supply voltage of 0.7V and optimizing with Dual-V_T at V_{DD} = 1.1V. Optimizing with Dual-V_T when the circuit is operating at voltages higher than 0.7V may not be advisable as it will be shown in the next section. Practical power savings are between 17% and 36% for the DVS range.

Dual-V_T optimization is expected to achieve larger power savings for larger circuits, as shown in [2]. The results show that the larger the operating voltage, the smaller the leakage power savings, this is due to the combination of two effects; the *drain-induced barrier lowering* (DIBL), which reduces V_T with larger supply voltages, and the sensitivity of leakage power to V_T, which is proportional to leakage power itself [2]. Increasing the supply voltage will reduce V_T and increase the leakage current due to DIBL effect as shown in [23]. Then the fixed difference of 0.1V for the zero-bias threshold voltage between high-v_T and low-v_T gates will offer different power savings depending on the final value of V_T due to DIBL effect.

The Figure also shows that optimizing at higher voltages offers larger power savings. (2) shows that the larger the supply voltage the smaller the impact of using high-v_T gates on gate delay. This implies that the higher the voltage, the more the room for assigning gates to high-v_T without producing timing violations. Then more leakage power savings can be achieved.

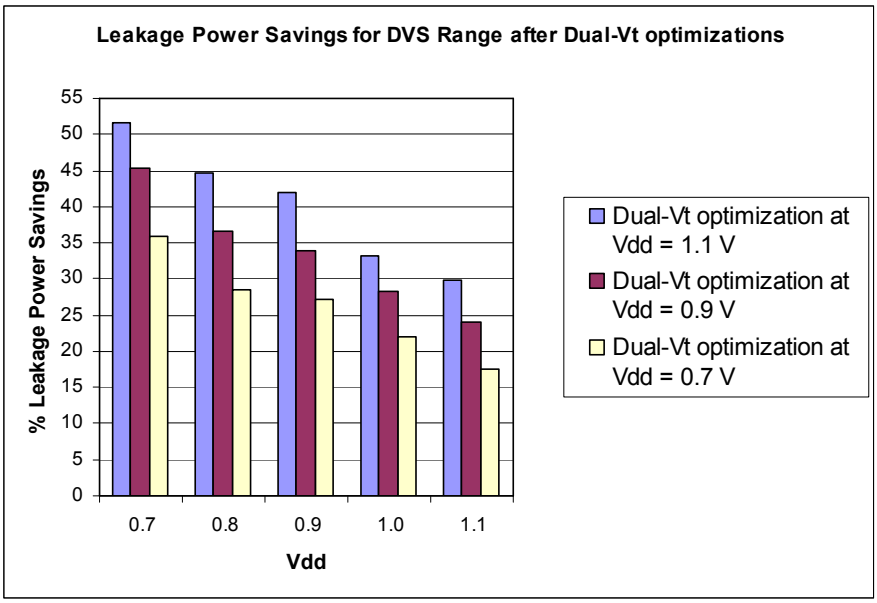

Fig. 14. Leakage power savings after applying Dual-V_T optimizations

5.3. DYNAMIC VOLTAGE SCALING IMPACT ON CIRCUIT OPTIMIZATIONS

To study the impact of DVS on circuit optimization, several HSPICE simulations were run sweeping the supply voltage (V_{DD}) through the DVS range for the original circuit after place and route, and after speed and leakage power optimizations at different operating voltages. The objective was to evaluate the risk of non-critical paths becoming critical paths at runtime, as presented in section 3, and to determine if there is any pattern that tells us how take DVS into consideration during circuit optimizations, in case that is necessary. Figs. 15, 16 and 17 show the response of the circuit to DVS after speed optimizations using logical effort. The Figures show the delay of all 15 paths in the set used for the case study. The set of paths was sorted with the circuit operating at the highest voltage, starting with number one, assigned to the longest path, to number

15, assigned to the shortest path. Fig. 15 shows that different paths scale differently while applying DVS. This relative scaling is consistent with the behavior predicted by Fig. 6 in the problem formulation section (section 3). Fig. 15 shows that initially (at 1.1 V) the set of paths is monotonically decreasing, but paths scale differently along with DVS and the set is not monotonically decreasing anymore. Fig. 16 shows that after speed optimizations, the relative delay variation (RDV) between paths is reduced in some cases and slightly increased in others. Figs. 15 and 16 show that although relative delay variations exist between paths, those RDV's are small for long paths (<7%) and do not represent any risk for non-critical paths becoming critical paths while applying DVS. Fig. 17 summarizes the results by showing only the highest and lowest voltages in the DVS range and the most relevant RDV's.

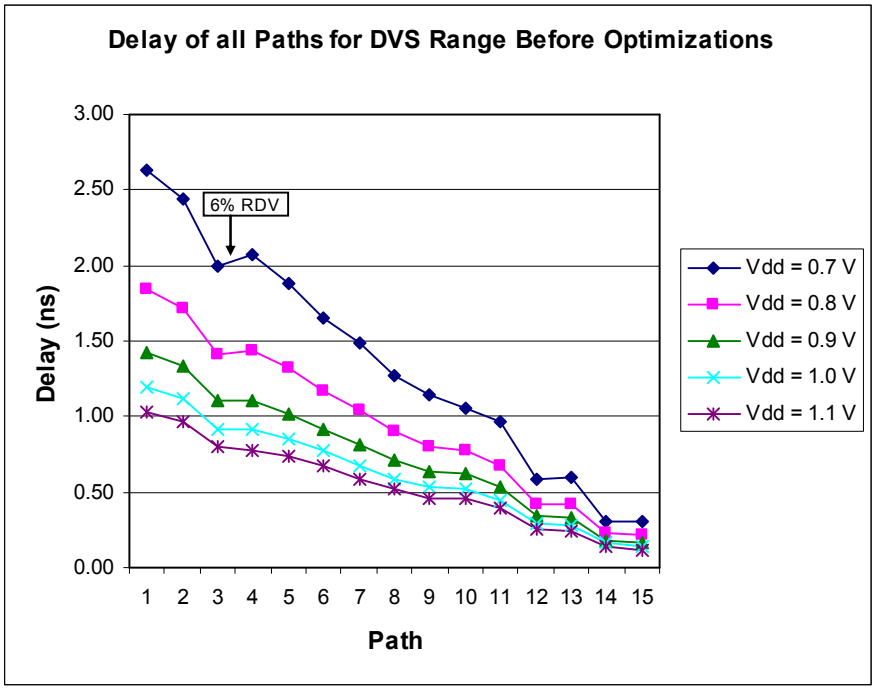

Fig. 15. Delay of all paths for DVS range before speed optimizations

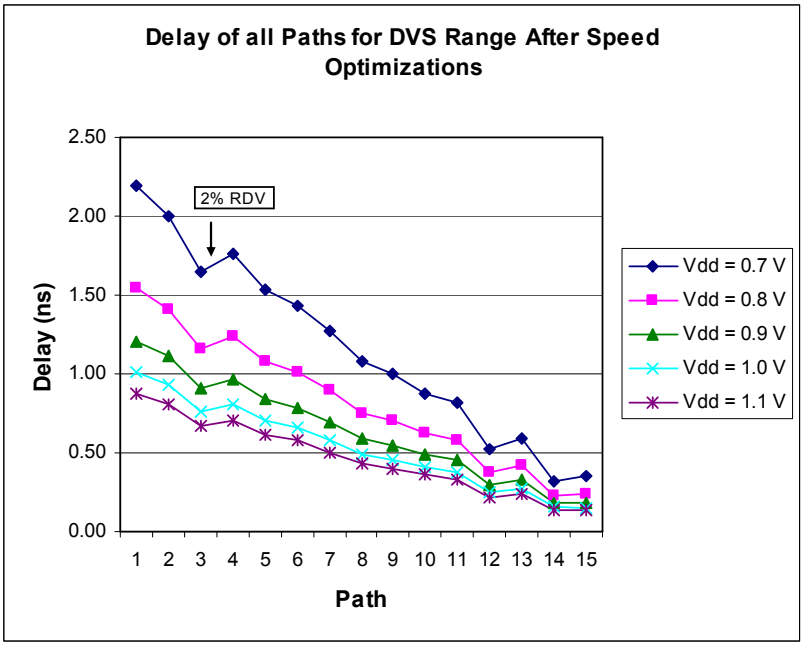

Fig. 16. Delay of all paths for DVS range after speed optimizations

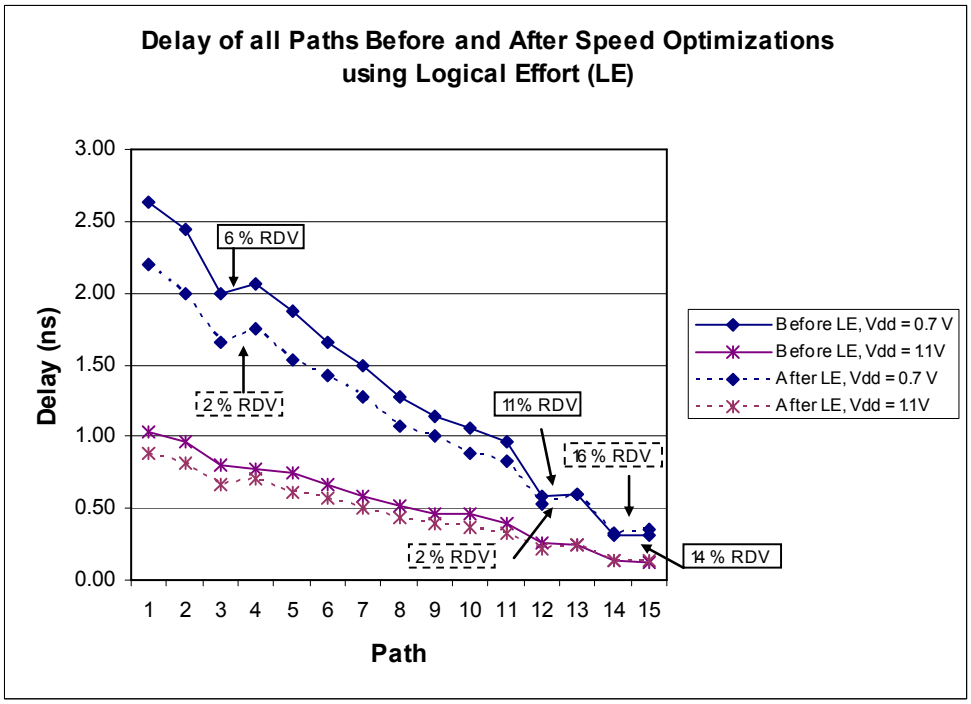

Fig. 17. Delay of all paths for DVS range before and after speed optimizations

Figs. 18 to 23 show the results of applying DVS to the circuit after both speed and leakage power optimizations performed to the circuit operating at the lowest, middle, and highest voltages in the DVS range. Figs. 18 to 20 present the same information as Figs. 15 to 17 did for speed optimizations, that is, they present the delay of all paths for the whole DVS range. The paths are sorted exactly the same as before, path #1 is the most critical path after place & route, and path #15 is the shortest path after place & route. Fig. 18 shows results after Dual-V_T optimizations for the circuit operating at 0.7V. The delay equalization due to the Dual-V_T optimization is clear in the curve for the circuit operating at 0.7V (curve in blue). There are 7 paths very close to the most critical path after the optimization. All other paths (path 8 to 15) are much shorter even after high-v_T assignments. From (3) and (6) we expect that the delay of high-v_T gates will scale up (become slower) at a higher rate with decreasing V_{DD} than those with low-v_T. Similarly, the high-v_T gate delays will scale down (become faster) at a higher rate with increasing V_{DD} than those with low-v_T. This behavior is observed in Fig. 18, where half of the curve is very flat (paths one to seven) due to the delay equalization at V_{DD} = 0.7V (blue curve). Increasing the supply voltage from 0.7V to 1.1V makes the delay of paths with high-v_T gates scale down faster than those paths without high-v_T gates (or with fewer high-v_T gates). This is observed as the flat section of the curve operating at 0.7V goes scaling down to the purple curve operating at 1.1V, where paths two to seven have scaled down apart from the most critical path, which doesn't have any high-v_T gate (due to the nature of Dual-V_T algorithms). All paths other than the most critical path have zero or more gates assigned to high-v_T, which means that

all paths scale down at a higher rate than the most critical path while sweeping up the supply voltage through the DVS range. This implies that the most critical path will always be the longest path in the circuit as shown in Fig. 18.

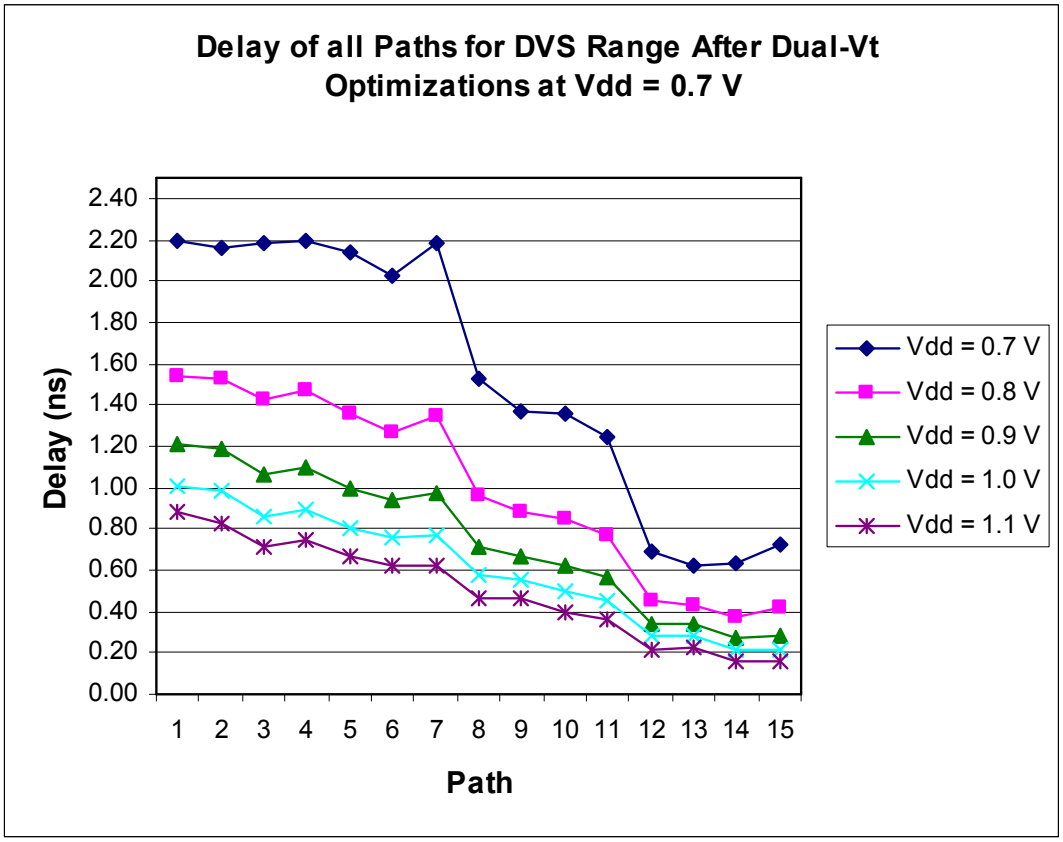

Fig. 18. Delay of all paths for DVS range after Dual-V_T optimizations at V_{DD} = 0.7V

Fig. 19 shows the results for Dual-V_T optimizations when the circuit is operating at 0.9V. Again, the delay equalization is clear for the green curve representing the circuit operating at V_{DD} = 0.9V. Paths with high-v_T gates scale down at a higher rate than the most critical path as in the previous case, that is for supply voltages between

0.9V and 1.1V. On the other hand, high-v_T paths will scale up at a higher rate for supply voltages lower than 0.9V. Fig. 19 shows that there is a problem with that, since non-critical paths 2, 3 and 4 have become critical paths while scaling the supply voltage, all of them have become longer paths than the original critical path at design-time (path 1). Fig. 20 shows that path 1 is not the most critical path anymore for operating voltages lower than 1.1V when optimizing with Dual-V_T at V_{DD} = 1.1V.

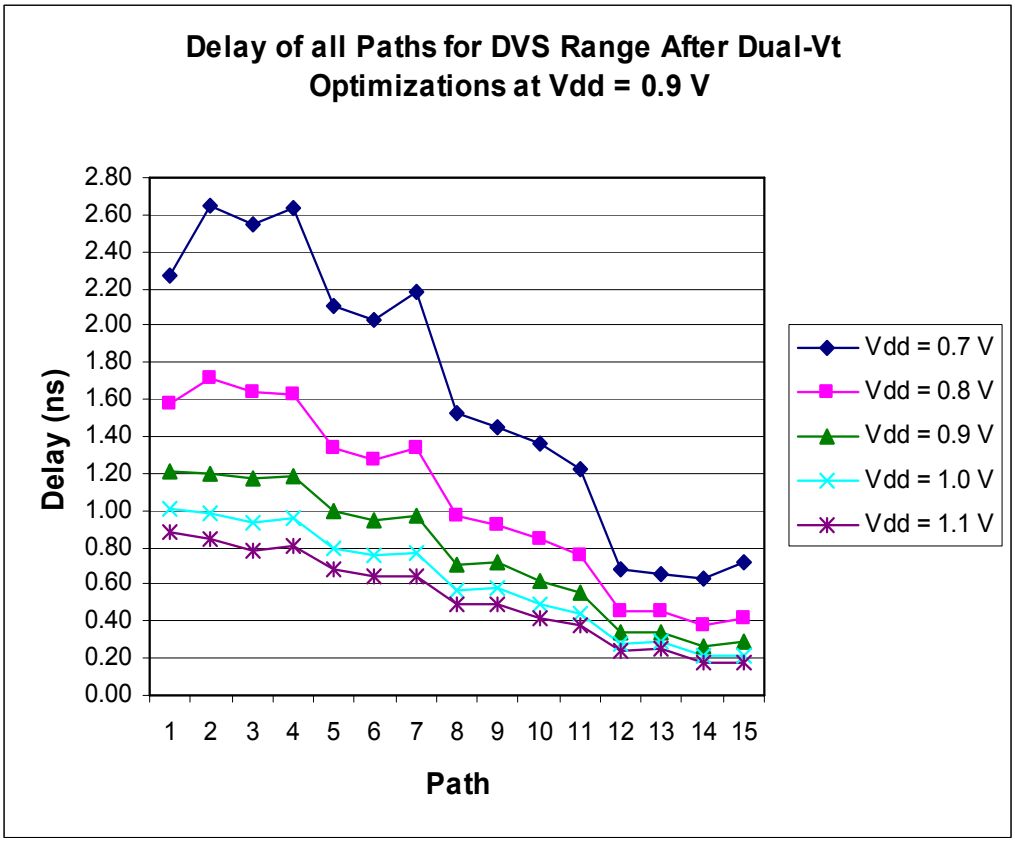

Fig. 19. Delay of all paths for DVS range after Dual-V_T optimizations at V_{DD} = 0.9V

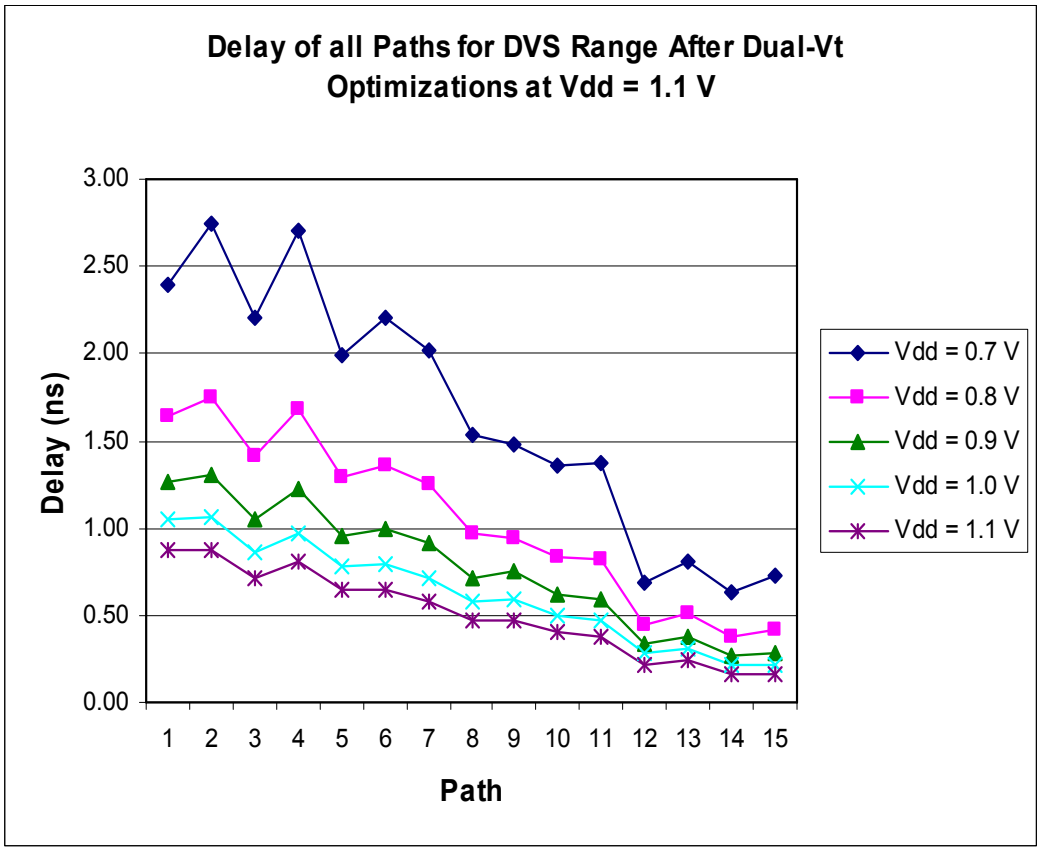

Fig. 20. Delay of all paths for DVS range after Dual-V_T optimizations at V_{DD} = 1.1V

The Dual-V_T approach as applied in this case study has the only purpose of reducing leakage power. Optimizing with Dual-V_T should have no impact on circuit delay; that is, the most critical path should be exactly the same before and after Dual-V_T optimizations. Moreover, the circuit speedup due to Dual-V_T optimizations should be zero at any operating supply voltage. That is a key concept for the algorithm as presented in [2]. Results presented in Figs. 18, 19, and 20 show that the circuit is actually experiencing a slowdown for some optimization cases as shown in Fig. 21.

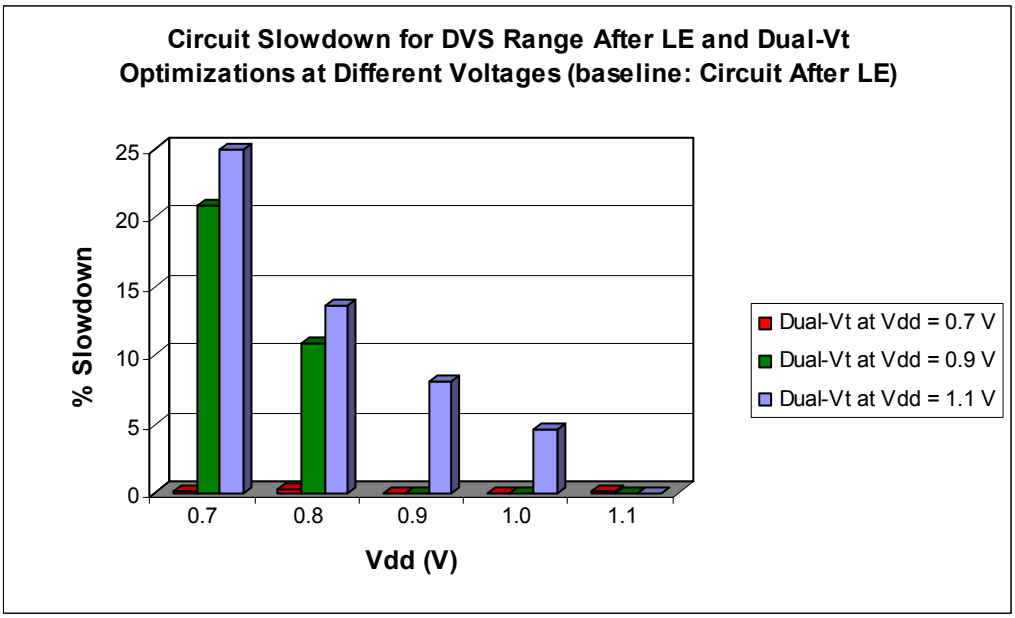

Fig. 21. Circuit slowdown for DVS range after speed and power optimizations

Fig. 21 shows that optimizing with Dual-V_T at V_{DD} = 0.7V is the only case that doesn't produce circuit slowdown. In the other two cases (optimizing with Dual-V_T at supply voltages of 0.9V and 1.1V) the circuit is slowed down as much as 25%. The slowdown is actually due to non-critical paths at design-time that turn into critical paths while sweeping the supply voltage to simulate DVS. By comparing Fig. 11 with Fig. 21 it is revealed that optimizing with Dual-V_T at supply voltages higher than 0.7V results in a circuit that is slower than the original circuit after place & route, before speed optimizations. For example, the 17% speed up achieved after logical effort optimization is overridden by the 25% slowdown when the circuit is operating at V_{DD}=0.7V and has been optimized with Dual-V_T at V_{DD}=1.1V.

Figs. 22 and 23 show the impact of DVS on Dual-V_T optimizations at different voltages. The Figures show the most relevant paths regarding to relative delay scaling. A good insight of the circuit's behavior through the DVS range is obtained by showing the delay of some paths after both speed and power optimizations, normalized to the delay of the critical path after speed optimizations only. The Figures show how different Dual-V_T optimizations present different scaling behavior while sweeping the supply voltage through the DVS range. The Figures clearly show that optimizing at a voltage higher than the lowest voltage in the DVS range results in non-critical paths at design-time turning into critical paths at run-time. This is due to the delay of high-v_T paths scaling up at a higher rate than the most critical path (without any high-v_T gates) during supply voltage reduction. The Figures also show, on the other hand, that when DVS is adjusting the voltage up, the delay of high-v_T paths scales down at a higher rate than the most critical path. This last scenario does not represent any problem since the critical path at design-time (path 1) remains as the longest path through the whole DVS range. Important and summarizing information provided by Fig. 22 is that non-critical paths become critical when the circuit is operating right below 1.1V and Dual-V_T optimizations have been performed at V_{DD}=1.1V. Similarly, Fig. 23 shows that non-critical paths become critical when the circuit is operating below 0.9V and Dual-V_T optimizations have been performed at V_{DD}=0.9V. Relative delay variations (RDV) as large as 32% are observed.

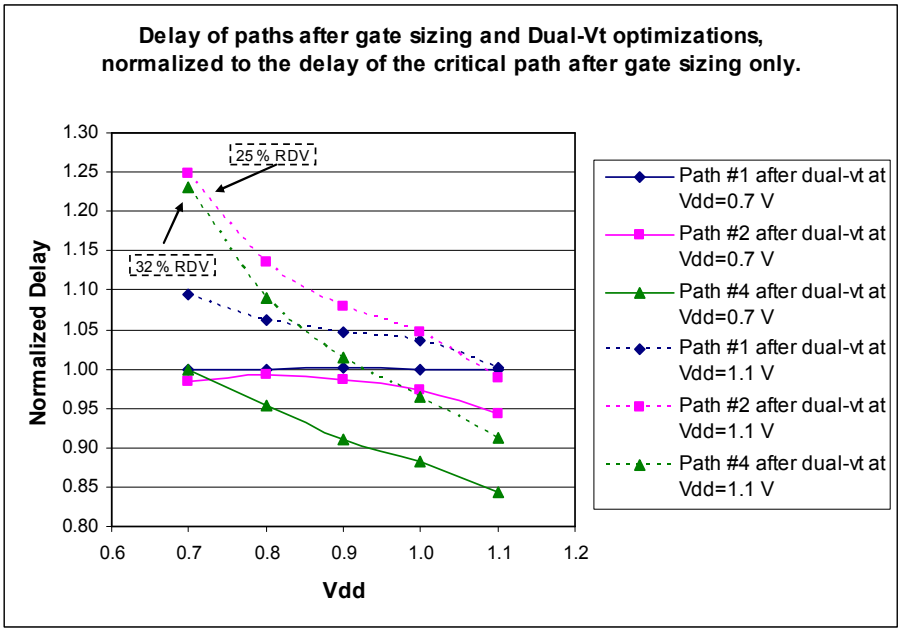

Fig. 22. Impact of DVS after Dual-V_T optimizations at the lowest and highest supply voltages in the DVS range

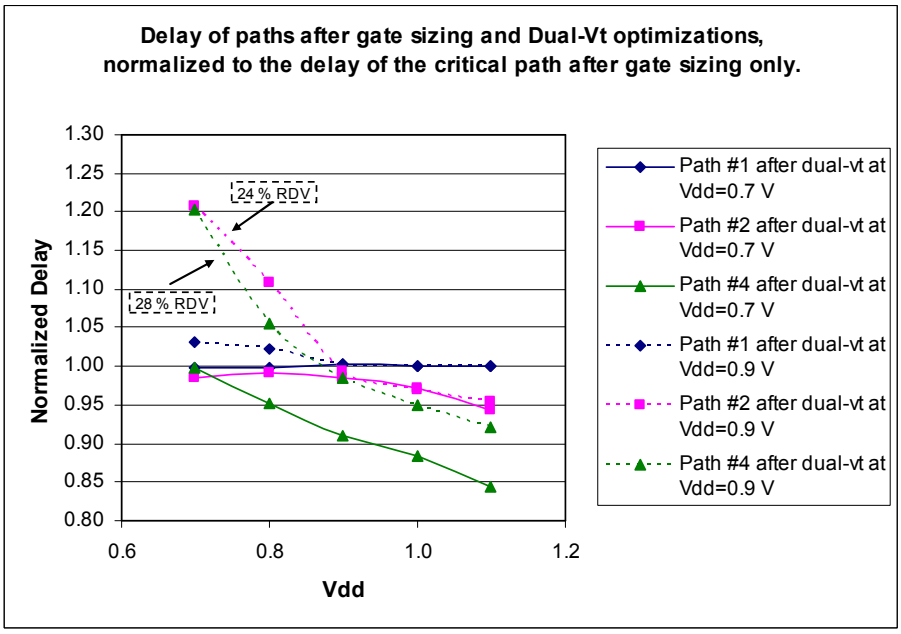

Fig. 23. Impact of DVS after Dual-V_T optimizations at the lowest and middle supply voltages in the DVS range

6. CONCLUSIONS AND FUTURE WORK

We studied the impact of dynamic voltage scaling on circuit optimizations for speed and leakage power. This case study was performed to the benchmark circuit ISCAS-85 c432 using a 45nm technology. Speed optimizations were performed using the method of logical effort as presented in [18]. Leakage optimizations were performed using a Dual-V_T algorithm based on [2]. Tools were implemented in C++ to automate and integrate circuit optimizations into the whole design flow, from synthesis to HSPICE simulations for DVS as shown in Fig. 7. The optimization tools were validated by HSPICE simulations. The speed optimization tool offered a maximum speedup of 17%. The power optimization tool offered as much as 35% leakage power savings.

The results showed that DVS had no impact on speed optimizations since paths close to the critical path presented small relative delay variations (RDV's < 3%) which does not represent a risk for non-critical paths to become critical paths at run-time while applying DVS. Larger relative delay variations were observed for the shortest paths, with RDV's as large as 16%, but those variations are not large enough as to turn the shortest paths into critical paths.

The results showed that DVS has impact on Dual-V_T optimizations, since non-critical paths at design-time became critical paths at run-time for optimizations performed at a supply voltage higher than the lowest voltage in the DVS range. This suggests that Dual-V_T optimizations should be performed at the lowest voltage in the DVS range, otherwise non-critical paths will become critical at run-time. Non-critical

paths became longer than the critical path during run-time by as much as 25%. One shorter path showed as much as 32% RDV while applying DVS and came close to becoming the most critical path at run-time. Over designing could be considered to overcome the situation of non-critical paths becoming critical paths at runtime. A 25% over design would be enough for this case study, but the fact that RDV's as large as 32% were observed, plus the possibility of having paths with much more complex structures scaling with much larger RDV's suggests that the required over design could be much larger than 25% for larger and more complex circuits with more complex path structures.

A set of tools were implemented to automate several tasks and integrate them into the flow of Fig. 7 so that it can be useful for future research. The circuit c432 used in this case study is a small design within the ISCAS-85 family of benchmarks. Hence, an extension to this work could be the use of the whole flow implemented in this thesis to study the impact of DVS on circuit optimizations for larger designs. In order to do that, some work has to be done regarding the Dual-V_T optimizations which were very time consuming since they are based on full HSPICE simulations. The options would be to either characterize the cells for dual-vt and different supply voltages, and program a static timing analyzer using lookup tables so that full HSPICE simulations are avoided, or migrate the C++ Dual-V_T optimization tool to UNIX, so that the full HSPICE simulations can be run on the supercomputers at the supercomputing facilities at Texas A&M University. Characterizing the cells for dual-vt and different voltages could be more time consuming than running full HSPICE simulations for the circuit used in this case study, but can be a good option for future research.

Wiring capacitances were not treated separately while performing speed optimizations with the method of logical effort. Those capacitances were taken into account as if they were part of the gate load being driven. This implies that the optimizations for speed are not optimal. Again, the circuit used in this case study is small and the wiring capacitances didn't dominate the total capacitance of the nets. Including wiring delay optimizations into the logical effort tool is another opportunity for future work.

REFERENCES

[1] M. Anis and M. Elmasry, *Multi-Threshold CMOS Digital Circuits: Managing Leakage Power*. 1st ed. Norwell, MA: Kluwer Academic Publishers, 2003.

[2] L. Wei, Z. Chen, K. Roy, M. Johnson, and V. De, "Design and Optimization of Dual-Threshold Circuits for Low-Voltage Low-Power Applications," *IEEE Transactions on VLSI Systems,* vol. 7, no. 1, pp. 16-24, March 1999.

[3] P. Pant, V. De, and A. Chatterjee, "Simultaneous Power Supply, Threshold Voltage, and Transistor Size Optimization for Low-Power Operation of CMOS Circuits," *IEEE Transactions on VLSI Systems*, vol. 6, no. 4, pp. 538-545, December 1998.

[4] V. Sundararajan and K. Parhi, "Low Power Synthesis of Dual Threshold Voltage CMOS VLSI Circuits," in *Proceedings of the IEEE International Symposium on Low Power Electronics and Design*, 1999, pp. 139-144.

[5] L. Wei, K. Roy, and C. Koh, "Power Minimization by Simulatenous Dual-Vth Assignment and Gate-sizing," in *Proceedings of the IEEE Custom Integrated Circuits Conference*, 2000, pp. 413-416.

[6] S. Sirichotiyakul, T. Edwards, O. Chanhee, Z. Jingyan, A. Dharchoudhury, R. Panda, and D. Blaauw, "Stand-by Power Minimization though Simultaneous Threshold Voltage Selection and Circuit Sizing," in *Proceedings of the 36th Design Automation Conference*, 1999, pp. 436-441.

[7] K. Roy, L. Wei, and Z. Chen, "Multiple-Vdd Multiple-V_{th} CMOS (MVCMOS) for Low-Power Applications," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, vol. 1, 1999, pp. 366-370.

[8] M. Khellah and M. Elmasry, "Power Minimization of High-Performance Submicron CMOS Circuits Using a Dual-V_{DD} Dual-V_{th} (DVDV) Approach," in *Proceedings of the IEEE International Symposium on Low Power Electronics and Design*, 1999, pp. 106-108.

[9] N. Weste and D. Harris, *CMOS VLSI Design: A Circuits and Systems Perspective*, 3rd ed. Boston, MA: Addison Wesley, 2005.

[10] S. Sirichotiyakul, T. Edwards, C. Oh, R. Panda, and D. Blaauw, "Duet: An Accurate Leakage Estimation and Optimization Tool for Dual-V_T Circuits," *IEEE Transactions on VLSI Systems – Special Issue on Low Power Electronics and Design*, vol. 10 no. 2, pp. 79-90, 2002.

[11] K.-S. Yeo and K. Roy, *Low-Voltage, Low-Power VLSI Subsystems*, 1st ed. New York, NY: McGraw-Hill, 2005.

[12] T. Burd and R. Brodersen, "Energy Efficient CMOS Microprocessor Design," in *Proceedings of the 28th Hawaii International Conference on System Sciences*, vol. 1, 1995, pp. 288-297.

[13] T. Pering, T. Burd, and R. Brodersen, "The Simulation and Evaluation of Dynamic Voltage Scaling Algorithms," in *Proceedings of the IEEE International Symposium on Low Power Electronics and Design*, vol. 1, 1998, pp. 76-81.

[14] K. Flautner, S. Reinhardt, and T. Mudge, "Automatic Performance Setting for Dynamic Voltage Scaling," *Wireless Networks*, vol. 8, no. 5, pp. 507-520, 2002.

[15] T. Burd, T. Pering, A. Stratakos, and R. Brodersen, "A Dynamic Voltage Scaled Microprocessor System," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 11, pp. 1571-1580, 2000.

[16] W. T. Ng and O. Trescases, "Power Management for Modern VLSI Loads Using Dynamic Voltage Scaling," in *Proceedings of the 7th International Conference on Solid-State and Integrated Circuits Technology*, vol. 2, no. 18, pp. 1412-1415, 2004.

[17] G. Yeon and M. Horowitz, "A fully Digital, Energy-Efficient, Adaptive Power-Supply Regulator," *IEEE Journal of Solid-State Circuits*, vol. 34, pp. 520-528, 1999.

[18] I. Sutherland, B. Sproul, and D. Harris, *Logical Effort: Designing Fast CMOS Circuits*, 1st ed. San Francisco, CA: Morgan Kaufmann, 1999.

[19] S. Karandikar and S. Sapatnekar, "Technology Mapping Using Logical Effort for Solving the Load-Distribution Problem," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 1, 2008, pp. 45-58.

[20] R. Gonzalez, B. Gordon, M. Horowitz, "Supply and Threshold Voltage Scaling for Low Power CMOS," *IEEE Journal of Solid-State Circuits*, vol. 32, no. 8, pp. 1210-1216, 1997.

[21] J. H. Stathis, "Reliability Limits for the Gate Insulator in CMOS Technology," *IBM Journal of Research and Development*, vol. 46, no. 2, pp. 265-286, 2002.

[22] M. Hirabayashi, K. Nose, and T. Sakurai, "Design Methodology and Optimization Strategy for Dual-Vth Scheme using Commercially Available Tools," in *Proceedings of the International Symposium on Low Power Electronics and Design*, 2001, pp. 283-286.

[23] S. Martin, K. Flautner, T. Mudge, and D. Blaauw, "Combined Dynamic Voltage Scaling and Adaptive Body Biasing for Lower Power Microprocessors under Dynamic Workloads," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, 2002, pp. 721-725.

[24] Electric's VLSI Design System Manual. Staticfreesoft, 2008. [Online]. Available: http://www.staticfreesoft.com/manual/text/chap09-12.html

[25] ISCAS-85 High-Level Models. Department of Electrical and Computer Science, University of Michigan, 2008 [Online]. Available: http://www.eecs.umich.edu/~jhayes/iscas.restore/benchmark.html

[26] 45nm FreePDK kit. VLSI Group, Oklahoma State University, 2008 [Online]. Available: http://vcag.ecen.okstate.edu/projects/scells/

VITA

Carlos Alberto Esquit Hernandez received his Bachelor of Science degree in electronics engineering from University of the Valley of Guatemala, Guatemala City, in June 2003.

He worked for Siemens Electrotecnica S.A. as an Automation Engineer from March 2004 to July 2006. He implemented industrial automation solutions for industries in Guatemala and Dominican Republic while working in that position.

He got a Fulbright grant sponsored by the US Department of State to pursue graduate studies in the United States. He used this grant to join the Department of Electrical and Computer Engineering at Texas A&M University in August 2006. He received his M.S. degree in Computer Engineering in May 2009. He will join the Department of Electronics at University of the Valley of Guatemala starting in January 2009, where he will be teaching undergraduate courses and working at the Research Institute of that University. Carlos Esquit may be reached at the Department of Electronics Engineering office J305 at University of the Valley of Guatemala, 18 Av. 11-95 Z. 15, V.H. III, Guatemala City, Guatemala. His email address is carlosesquit@gmail.com.